PATCHWISE SPARSE DICTIONARY LEARNING FROM PRE-TRAINED NEURAL NETWORK ACTIVATION MAPS FOR ANOMALY DETECTION IN IMAGES

Anonymous authors
Paper under double-blind review

ABSTRACT

In this work, we investigate a methodology to perform anomaly detection and localization in images. The method leverages both sparse representation learning and the adoption of a pre-trained neural network for classification purposes. The objective is to assess the effectiveness of the K-SVD sparse dictionary learning algorithm and understand the role of neural network activation maps as data descriptors. We extract meaningful representation features and build a sparse dictionary of the most expressive ones. The dictionary is built only over features coming from images without anomalies. Thus, images containing anomalies will either have a non-sparse representation as linear combinations of the dictionary elements or a high reconstruction error. We show that the proposed pipeline achieves state-of-the-art performance in terms of AUC-ROC score over benchmarks such as MVTec Anomaly Detection, Rd-MVTec Anomaly Detection, Magnetic Tiles Defect, BeanTech Anomaly Detection Datasets, Kolektor Surface Defect Dataset.

1 INTRODUCTION

Anomaly detection is the problem of identifying, given a set of data, those examples that deviate from the expected or standard behavior. In computer vision, anomaly detection tasks recognize those images showing objects or patterns with corrupted components or irregularities. Its applications range from industrial quality control to medical analysis. The typical approach to anomaly detection tries to model data distribution without anomalies and then estimate irregularities as examples with lower probability outcomes. The task is particularly demanding because it is difficult to understand and model anomaly occurrences due to three main challenges: *data imbalance*, *concept drift*, and *selection bias*. The first relates to the lack of anomalous examples; the second represents the variation of anomalies over time; the third reveals that the partial knowledge of the phenomenon introduces a bias in acquired data.

These reasons pushed researchers to focus on unsupervised or weekly supervised machine learning techniques, leveraging part of the available data, usually composed of defect-free examples. Indeed especially in the context of anomaly detection over images for industrial quality control, we face data imbalance and concept drift problems, but we usually can collect sets of high-confidence level normal images. The variety of methods for unsupervised anomaly detection is wide and diversified; the following works give a comprehensive review of existing solutions: Pimentel et al. (2014), and Ehret et al. (2019). In this section, we restrict ourselves to a brief overview of a specific subset of them, more recent and closer to our work.

In recent years, at the 2019 International Conference on Pattern Recognition (ICPR), a new dataset containing few thousand images of objects and textures with and without anomalies has been released by MVTec Software GmbH company (Bergmann et al., 2019). The purpose has been to provide the scientific community with a dataset to address real-world industrial scenarios, compare recent techniques, and understand the overall scientific progress over the topic.

At the same time, deep learning and artificial neural networks based models have had a significant impact on computer vision, leading to improvements in image classification, segmentation, and object detection, e.g. Krizhevsky et al. (2012), Simonyan & Zisserman (2014), Xie et al. (2017), He et al. (2016), Szegedy et al. (2015). Many of these techniques have also been proposed for anomaly detection. In the article cited above (Bergmann et al., 2019), authors have also proposed a summary and an evaluation of these techniques. They start analyzing reconstruction-based models such as Convolutional Autoencoders (Goodfellow et al., 2016). This generative neural network based on convolutional layers compresses the image and generates a reconstruction. The network is trained

over non-defective images with classic L2 or with Structural Similarity Measure as loss function (Bergmann et al., 2018). High reconstruction errors should characterize images with anomalies. Authors then investigate a model based on Generative Adversarial Networks (Goodfellow et al., 2014). This is a two-networks model, a Generator and a Discriminator, trained simultaneously with a two-player min-max optimization process. The Generator learns a distribution on healthy samples from a latent space to a manifold trying to fool the Discriminator. The latter computes the likelihood of a given sample to be a real or fake image. The assessed GAN-based model was the one proposed in Schlegl et al. (2017), named ANOGAN, appositely designed for the anomaly detection task.

Authors of Bergmann et al. (2019) continue the analysis considering traditional feature-based methods such as Gaussian Mixture Model (Böttger & Ulrich, 2016) based on hand-crafted feature descriptors from defect-free images.

Another important methodology assessed is a Convolutional Neural Network self-similarity method proposed by Napoletano et al. (2018). The interesting point of this latest work is to use clustered feature descriptions obtained from the *activation maps* of a ResNet-18 (He et al., 2016) pre-trained over ImageNet dataset (Deng et al., 2009). Convolutional Neural Networks usually have various layers, and intermediate outputs are identified as features maps, activation maps, or simply activations. In Napoletano et al. (2018), training features are extracted from patches that are cropped from the input images. First, activations are dimensionally reduced with a Principal Component Analysis; then, a K-Means algorithm calculates k clusters and relative centroids. These are used as atoms to build a dictionary. Normal images coming from a part of the validation set are then used to estimate the mean and variance of the euclidean distance from most similar dictionary elements. Boundaries of the estimated Gaussian distribution are used at inference time as a threshold. If the Euclidean distance to the most n similar dictionary elements is higher than the threshold, they flag the patch as anomalous.

Starting from the idea of Napoletano et al. (2018), different authors have investigated the adoption of pre-trained neural network activation maps to model distributions over images without anomalies. In particular, the authors of Defard et al. (2020) (PaDiM) show the effectiveness of modeling each patch activation map as Multivariate Gaussian Standard distributions, considerably improving the results presented in Bergmann et al. (2019) in terms of per-pixel and per-image scores. Other papers leverage the same idea: Cohen & Hoshen (2021), Rippel et al. (2020), Yang et al. (2020), Yi & Yoon (2020), Roth et al. (2021), Li et al. (2021). Their methodologies differ in the pre-trained architecture they have chosen, how they have aggregated information from different layers, estimated the distribution, and calculated the anomaly score. Moreover, to achieve high-performance scores, they usually operate class by class, i.e., in a one-class learning scenario. The work proposed in Roth et al. (2021), named PatchCore, builds a maximally representative memory bank of nominal patch-features and then applies a core set-reduction method to increase efficiency. Anomaly detection is then performed at a patch level with distance from the closest memory bank element.

Other works, as Pirnay & Chai (2021), involve specific techniques, such as inpainting, to obscure certain parts of the image and teach the network to reconstruct the missing part according to what is present in anomaly-free examples. The methodology presented in Zavrtanik et al. (2021) shows a full multiscale inpainting strategy to force the reconstruction of subregions of the image. Similarly, Wang et al. (2021) uses a student-teacher framework, a technique closer to GAN.

Our contribution is a continuation of the original work of Napoletano et al. (2018), and it assesses the application of sparse learning representation methods. While in Napoletano et al. (2018) the authors build the dictionary out of a PCA and K-Means of latent representation features, we propose a more structured approach based on K-SVD. This is a well-known sparse dictionary learning algorithm effectively used to address image denoising and anomaly detection tasks (Aharon et al., 2006). The algorithm is used to learn a dictionary describing the essential channels features for each spatial element of the activation maps. While there has been wide adoption of dictionary learning and transfer learning for classification purposes in literature (Maurer et al., 2013; Gu & Li, 2021; Tang et al., 2020; Zarka et al., 2019), we are the first to adopt them in an unsupervised scenario for anomaly detection

The proposed pipeline results to be more robust than other methods, allowing to drop the one-class learning scenario and managing a more significant variance at one time. This is a fundamental aspect of real industrial world applications: products can vary in size and shape, and data gathered might not be sufficient for all sub-product; building a different model for each sub-product might also be expensive in terms of time and memory. Results show that performance is competitive with respect to state-of-the-art and confirm that features extracted from robust classification networks can be leveraged to address other tasks in computer vision too.

This paper is organized into a total of 4 sections. We have presented an introduction to the problem and described related solutions in Section 1. Section 2 describes in detail the methodology adopted. Section 3 briefly describes the conducted experiments in terms of ablation studies and comparison with state-of-the-art models. Section 4 discusses the results and concludes by highlighting possible future development for the described methodology. The appendix contains further experiments conducted to assess the impact of K-SVD main hyper-parameters and the best possible configuration.

2 METHOD

This section presents how to combine the features extracted from a pre-trained network and a sparse dictionary learning procedure in an end-to-end pipeline for anomaly detection over images. The goal of the learning method is to obtain a dictionary that provides a sparse representation along the channels dimension for each spatial element of the activation maps. Because this spatial element corresponds to a *receptive field* in the original image, the method retain the most relevant features extracted by the network of each image patch. We underline that the method does not build a single dictionary: but a collection of dictionaries, one for each receptive field (or patch) of the selected layer embedding.

The learning step is done through a *sparse coding* approach. Sparse coding or sparse dictionary learning is a representation learning method which aims at finding a sparse representation $\boldsymbol{X}$ of the input data $\boldsymbol{S}$ in the form of a linear combination of basic elements of a dictionary $\boldsymbol{D}$, namely atoms. To give a more formal description, we define $\boldsymbol{S} \in \mathbb{R}^{m \times n}$ to be a matrix representing a collection of signals, with $n >> m$. Let $\boldsymbol{D} \in \mathbb{R}^{m \times m}$, $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ be respectively the dictionary and the sparse representation of the signals. We then define the following problem, known as *simultaneous sparse approximation*:

$$\arg\min_{D,X} \frac{1}{2}\|DX - S\|_F^2 + \lambda\|X\|_1 \text{ such that } \|d_k\|_2 = 1 \forall k. \tag{1}$$

In Equation 1, $\|\cdot\|_F^2$ is the square of the Frobenius norm, $\lambda > 0$, $\|\cdot\|_1$ is the L_1 norm (in some formulation we can find the L_0 norm), and $\boldsymbol{d}_k$ are the columns of $\boldsymbol{D}$. The formulated problem aims to simultaneously solve the task of learning a representative set of elements from a set of signals (the dictionary) and a sparse representation of them. The latter should retain the most important features of the signal.

2.1 TRANSFER LEARNING FROM PRE-TRAINED CONVOLUTIONAL NEURAL NETWORKS

Our patchwise sparse dictionary learning algorithm leverages Transfer Learning. Pre-trained Convolutional Neural Networks activation maps have proven to be effective features descriptors in different pattern recognition tasks: Shin et al. (2016), Kandaswamy et al. (2014), Raina et al. (2006), Bergman & Hoshen (2020). In particular, as highlighted in the introduction, pre-training over the Imagenet dataset can help build descriptors for modeling normal data distributions.

In our work, we have selected architectures that achieve state-of-the-art classification error rate over Imagenet (such as ResNet He et al. (2016), or EfficientNet Tan & Le (2019)). We have also chosen the output of specific layers, rescaling them with an average pooling operation to match different spatial dimensions (Rippel et al., 2020). Each channel of this output contains spatial representations of input image patches at a specific network depth.

As done in Defard et al. (2020), we studied how to effectively combine different layers activation maps to maximize the detection capability of the pipeline. In our case, there is a constrain to the total number of layers output we can select; indeed, solving a dictionary learning problem usually goes through solving an undetermined linear system problem, and the total number of dictionary atoms can not exceed the total number of examples. This means we can not select architectures or layers with a number of channels greater than the total number of examples. Moreover, building a comprehensive dictionary is memory and time-consuming. If dimensions grew indefinitely, also subsequent calculations done with this dictionary would request impractical times.

At the start of our pipeline, we pass all the images of the training dataset into the neural network and collect the intermediate outputs. We then build a tensor with the following dimensions: number of examples, number of channels, height, and width; the last two dimensions correspond to those of the activation maps closer to the network's input.

2.2 SPARSE DICTIONARY LEARNING WITH K-SVD

For each spatial element belonging to the collected outputs, we build a dictionary using the K-SVD algorithm (Aharon et al., 2006). In a general scenario, this algorithm tries to build from a set of signals a dictionary that sparsely represents each signal used to learn it. K-SVD is divided into a two-step procedure, where it alternates the signal sparse coding computation with a dictionary update performed under the conditions of preserving the just obtained sparse representation. While the first sparse coding computation can be performed leveraging algorithms such as Orthogonal Matching Pursuit (OMP, Cai & Wang (2011)) or Iterative Soft Thresholding (ISTA, Daubechies et al. (2004)), the dictionary learning step consists of updating each atom of the dictionary, computing a singular value decomposition, and taking the largest singular value. While classic resolution methods of this problem, such as Engan et al. (1999), alternate between updates of the coefficients and dictionary, K-SVD, at the $k - th$ iteration, restricts the update of the residuals to those columns corresponding to non-null coefficients of the considered atom.

As underlined above, this algorithm is run for each spatial location. This means that, in the end, we have a dictionary for each patch of the input image. Let $\boldsymbol{s}_z$ be a vector with number-of-channels elements depending on the spatial point z of a $w \times h$ feature maps; then the dictionary $\boldsymbol{D}_z$ is a channels $\times$ channels matrix, and the coefficients $\boldsymbol{x}_z$ are as well a channels-sized vector. Finally, the total number of dictionaries is equal to the product of the spatial dimensions of the activation maps. Dictionary dimensions are kept to be as small as possible to save time and memory costs. It is worth noticing that this kind of approach might suffer from a non-aligned-images dataset: while patches look at the same image portions, portraited objects could change positions, increasing the patch variance. Still, the conducted experiments reveal a good level of robustness to such variations.

In our work, we adopted Orthogonal Matching Pursuit as the sparse coding algorithm, with a maximum sparsity level as stopping criteria. Our research has tested different initialization strategies for the dictionary, different sparse coding levels for the OMP algorithm, and different iterations numbers for the dictionary learning algorithm. These experiments are discussed in details in the Appendix Sections A.1, A.2, A.3.

2.3 THE ANOMALY DETECTION PIPELINE

Once we have selected a spatial point z of the embedding and retrieved the learned dictionary, we use it to calculate the $L1$ norm of the coefficients $\boldsymbol{x}_z$ and the reconstruction error defined with the Euclidean distance between $\boldsymbol{D}_z\boldsymbol{x}_z$ and the signal $\boldsymbol{s}_z$. We estimate the Gaussian bi-variate distributions of these two quantities on the training set, ending with a mean vector and a covariance matrix. The idea is that anomalous patches either present a high $L1$ norm, which means their representation is not sparse, or have a high reconstruction error. At the end of this learning procedure, we are left with different two-dimensional mean vectors and 2×2 covariance matrices equal to the number of dictionaries.

At testing time, we retrieve the encoding making a forward-pass in the network for a single example of a class. We compute the sparse representation with respect to the learned dictionary for each spatial point, applying the OMP algorithm. We then calculate the $L1$ sparsity level, the reconstruction error, and the Mahalanobis distance between the distribution of the estimated errors and those calculated over the samples. The Mahalanobis distance, i.e. the anomaly score, is normalized according to the class the example belongs to. Since this computation provides an anomaly map with width and height equal to those of the activation maps, and these usually have smaller spatial dimensions than the original input image, we apply a bicubic interpolation and a Gaussian smoothing (as done in Defard et al. (2020)) to extend the anomaly map spatial dimensions. We also aggregate the score taking the maximum value of the anomaly map to obtain an image-level anomaly score. The area under the receiving operating characteristic curve (AUC-ROC) is the metric adopted to evaluate this pipeline.

The proposed method leverages the powerful representation of Imagenet pre-trained convolutional neural networks in combination with an efficient encoding of the relevant information contained in image patches. While this method is close to those discussed in the introduction, there are several significant differences. Indeed, to our knowledge extent, we are the first to adopt a sparse dictionary learning algorithm (K-SVD) to manipulate features extracted from a network. Moreover, while approaches presented in the introduction do not directly deal with dimensionality reduction of chosen features in a structured way, we develop the pipeline around the sparse representation concept. Finally, our methodology does not operate class by class, as for the literature; it works instead for the dataset as a whole, tolerating a much wider variance. From a theoretical perspective, this is a crucial point: because the problem we are tackling is essentially an undetermined system, using more examples gives us the chance to choose activation maps with more features and thus have a

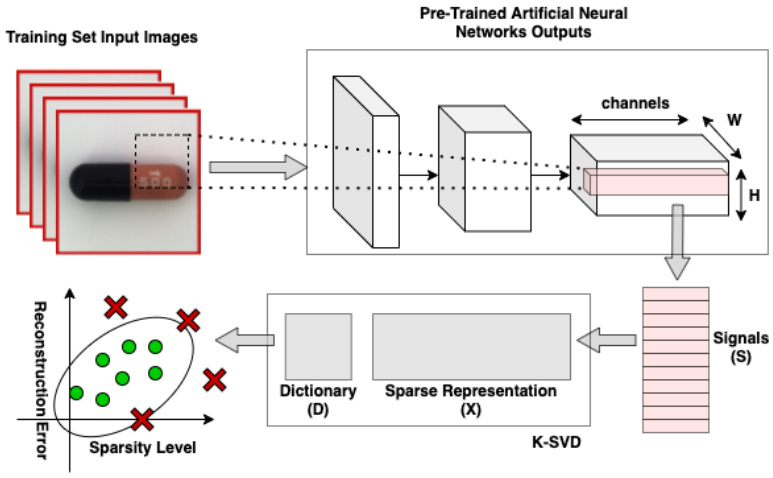


Figure 1: A diagram illustrating the proposed anomaly detection pipeline

greater representation power. With our method, a comparison between one class model vs all-classes model, on the MVTec Anomaly Detection Dataset, would only be possible using networks with few channels in the activation maps: we do not have enough examples for each class to adopt deeper layers. As it is shown later, this would noticeably lower the performance.

The proposed method could have a significant impact from another perspective: in the deployment of anomaly detection methods to actual industrial applications, building a model for each product sub-type might be unfeasible and could prevent the approach from scaling. A final overview of the described pipeline is presented in Fig. 1.

3 Experiments

The method described in previous sections depends on three key elements: the activation maps selection, the dictionary learning procedure, and the sparse representation of the training dataset elements. We have conducted different ablation studies to understand the impact of these factors on the proposed anomaly detection process.

For experiments of Sub-section 3.1.1, and those in the Appendix A.1, A.2, A.3, we have chosen Imagenet pre-trained ResNet-18, as done in Napoletano et al. (2018), and Defard et al. (2020). We use the pre-trained version of this network available through Pytorch APIs (Authors, Accessed 01-August-2021). Pytorch was also used for whatever concerned deep learning components. SparseLandTool library (Herzog, 2021) served instead as the core for OMP and K-SVD algorithms.

The ResNet-18 architecture is a sequence of three-operations blocks: convolution, batch normalization, and relu activation. This block is referred to as *convolution*, and it is repeated twice to form a *ResNet block*. Another fundamental characteristic of this network is the presence of *skip connections*, i.e., connections from previous to subsequent layers, able to jump those in the middle.

From a spatial perspective, the first sub-block inside each ResNet block keeps the exact spatial dimensions as the input, while the second halves them. ResNet blocks are repeated several times and sequentially stacked, doubling the convolutional filters at each new block. Activation maps dimensions are 56x56x64 for layer 1, 28x28x128 for layer 2, 14x14x256 for layer 3, and 7x7x512 for layer 4.

We tested the method described above over the MVTec Anomaly Detection Dataset. This dataset comprises 15 categories with 3629 images for training and 1725 images for testing. The training set contains only images without defects. The test set contains both defect-free images and anomalous ones. Five categories of the dataset cover different types of regular (carpet, grid) or random (leather, tile, wood) textures, and the other ten categories represent different types of objects. Objects might be rigid with a fixed appearance (bottle, metal-nut), while others are deformable (cable) or include variations (hazelnut). The test images of anomalous samples contain various defects on the object's surface (scratches, dents), structural anomalies like distorted object parts, or defects that manifest themselves by the absence of certain object parts.

Images have been loaded into a range of [0, 1] and then normalized using mean = [0.485, 0.456, 0.406] and standard deviation = [0.229, 0.224, 0.225]. Since images vary in pixel resolutions, they

Table 1: Layers selection results

	1^{st} **Layer**	2^{nd} **Layer**	3^{rd} **Layer**	4^{th} **Layer**	**1+2+3**	**1, 2, 3**
Textures classes	0.7606	0.881	0.9486	0.8376	0.9438	**0.9614**
	0.711	0.8588	0.9304	0.7968	0.9122	**0.9428**
Objects classes	0.7534	0.811	0.9134	**0.9204**	0.8562	0.9126
	0.717	0.8912	0.9716	0.9572	0.9538	**0.9772**
Total	0.757	0.846	0.931	0.879	0.900	**0.937**
	0.714	0.875	0.951	0.877	0.933	**0.960**

have been resized to (256, 256) and center-cropped to (224, 224), as performed in Defard et al. (2020).

3.1 Ablation Studies

Ablation studies over the K-SVD algorithm are reported in the appendix; we preferred to highlight the most interesting due to the limited amount of space. Nevertheless, results presented in the appendix show that best performance is reached when K-SVD is run with the dictionary initialized with DCT basis for 20 iterations, setting the OMP sparsity level to a quarter of the dictionary elements. We have adopted these settings in all the other experiments. Experiments have been run multiple times (2-3) to check the results' consistency. No fluctuations have been observed.

3.1.1 Layers Selection

This sub-section presents the experiments conducted to understand which are the most suitable layers and related activation maps. We have chosen ResNet-18 layers 1, 2, 3, 4 and run the pipeline described above. Spatial dimensions of activation maps differ from layer to layer, leading the number of dictionaries and the number of elements of the dictionaries to change accordingly. As done in Defard et al. (2020), we have also tested the combination of different activation maps in two ways: either before the dictionary calculation, aggregating layers 1, 2, and 3 as done in Rippel et al. (2020), or simply aggregating the anomaly detection maps created using the different dictionaries. In the first case, higher resolution spatial maps are average-pooled to have dimensions consistent with the smallest one; in the second case, different anomaly maps (each with same spatial dimensions, because we have interpolated them) are summed up.

Tab. 1 shows results obtained for different layers selection. The table reports result with an anomaly detection pipeline run with activation maps coming from Layer 1, Layer 2, Layer 3, Layer 4, and differently combined activation maps both in the calculation (1+2+3) and in the dictionary learning phase (1, 2, 3). The first number in each cell is the per-image AUC-ROC, while the second is the per-pixel AUC-ROC. We note that the score increases as the depth and number of the layers of filter increase until layer 4. The single-layer top performer is number 3; the subsequent layer has registered a drop instead, probably due to the lower activation maps resolution. As later experiments confirm, activation maps have to keep a minimum resolution of 14x14.

Interestingly, the best scores are achieved with the combination of activation maps coming from different layers. Nevertheless, these benefits are realized only when the different activation maps are used in the dictionary learning process (last column). The features extracted at different depths of the network capture different details that significantly increase the scores even if coarsely represented (through the average-pooling operation).

3.1.2 Architectures Selection

Results from Sub-section 3.1.1 have highlighted the link between the activation maps depth level and the dictionary learning process in the anomaly detection task. They have also highlighted a significant improvement when we combine different levels of ResNet-18. This sub-section investigates the possibility of using different deeper architectures while keeping the spatial resolution fixed. In other words, as done in Defard et al. (2020), we have tested the proposed pipeline with Wide ResNet-50, EfficientNet-B5, EfficientNet-B6, and EfficientNet-B7. We have selected the deepest activation map with 14x14 as spatial resolution (layer 3 of ResNet-18). For Wide-ResNet-50, layer 3 output channels grow from 256 to 1024. We thus have used a larger number of channels in the dictionary learning process.

For EfficientNet-B5, B6, and B7, we have used outputs of the 26, 30, and 37 mobile inverted bottleneck layers, denoted here as MBConv. This is the EfficientNet main building block (Tan et al.,

Table 2: Architectures selection results

	RN-18	**WRN-50**	**EN-B5 26**	**EN-B6 30**	**EN-B7 37**
Textures classes	0.9486	0.9798	0.9854	**0.9922**	0.9888
	0.9304	0.9554	0.943	0.949	**0.9576**
Objects classes	0.9134	0.9342	0.9206	0.9198	**0.9432**
	0.9716	**0.9726**	0.947	0.955	0.9404
Total	0.931	0.957	0.953	0.956	**0.966**
	0.951	**0.964**	0.945	0.952	0.949

2019), (Sandler et al., 2018). As said before, they all have the same spatial dimensions while slightly different channels numbers (176, 200, 220). The main difference is the layers' depth and the subsequent different representation power of the architectures.

Wide-ResNet-50 pre-trained model has been taken from Pytorch API, while for the EfficientNet-Bx networks, we used the implementation available at Melas-Kyriazi (Accessed 01-August-2021). Results of the experiments are shown in Tab. 2. The first number in each row is the per-image AUC-ROC, while the second is the per-pixel AUC-ROC.

We immediately notice how the adoption of layer 3 of Wide-ResNet50 causes a relevant performance improvement with respect to the best model from Section 3.1.1. The number of channels has dramatically increased, causing the dictionary learning procedure to be much slower ($\approx$ 6000 seconds per K-SVD iteration) with respect to the other networks ($\approx$ 200-300 seconds). This is primarily due to the high number of channels in comparison to the available examples. The learned dictionary is bigger and full of relevant features for describing standard behavior.

The most important thing to underline is that, while there is a small but significant difference between Wide-ResNet-50 model results and the ones based on EfficientNet, the number of channels used in latter cases is five times smaller. This means that in those situations with few available examples in the training set, we can easily trade channels dimension with the depth of the architecture without compromising the effectiveness of the anomaly detection pipeline. In other words, we can either choose a layer coming from a deeper network but with fewer channels or a layer from a shallow network with many channels in the activation maps. Moreover, as proved by experiments conducted in Section 3.1.1, combining different layers in the dictionary learning process can positively affect the scores.

3.2 Best Model and comparison with SOTA

Previous studies have shown how specific aspects of the proposed pipeline are relevant to the anomaly detection task. While experiments of Appendix A.1, A.2, and A.3 have given information about the dictionary learning procedure, experiments of Sections 3.1.1 and 3.1.2 show the impact of deeper layers activation maps and architectures. These results have given us insight on how to calibrate the best models for comparison with state-of-the-art methods over MVTec Dataset, Rd-MVTec AD Dataset, Magnetic Tile Defects Dataset (Huang et al., 2020), BeanTechAnomaly Detection Dataset (Mishra et al., 2021), and Kolektor Surface Defect Dataset (Tabernik et al., 2019). In this sub-section, we have adopted the standard pipeline configuration. K-SVD algorithm is run for 20 iterations with dictionaries initialized with DCT basis, and OMP sparsity level set to a quarter of the activation maps channels. The only thing that changes from one experiment to another is the type of architecture and the subsequent layers producing the activation maps.

3.2.1 MVTec Dataset and Non-Aligned MVTec Dataset

For our final experiment on the MVTec Dataset, we have chosen to adopt EfficientNet-B6 architecture, selecting outputs of layers MVConv Block 15, 22, 30. This choice is motivated both by the experiment of Sub-sections 3.1.2, and 3.1.1. Thanks to the EfficientNet configuration, we can select different layers at different depths but with the same spatial resolution. We could have adopted Wide-ResNet-50 architecture (and try to combine layers 1, 2, and 3), but we preferred to sacrifice a little bit of accuracy in exchange for a faster pipeline.

We have reported here the scores of the state-of-the-art anomaly detection methods for a fair comparison taken from Roth et al. (2021). The first number in each row is the per-image AUC-ROC, while the second is the per-pixel AUC-ROC. The pre-trained neural network used in PaDiM method is Wide-ResNet-50 for pixel-level detection and EfficientNet-B5 for image-level detection. Wide-ResNet-50 has also been adopted for SPADE method. The version of PatchCore adopts a memory bank level subsampling of 25%. Results shown in Tab. 3 demonstrate the ability to match state-of-

Table 3: Comparison with related methods over MVTec Dataset

	SPADE	PatchSVDD	Mahal. AD	PaDiM	PatchCore	PSDL EN-B6 (ours)
Total	0.855	0.921	0.958	0.979	**0.991**	0.962
	0.96	0.957	-	0.975	**0.981**	0.962

Table 4: Comparison with related methods over Rd-MVTec Dataset

	VAE	SPADE	PaDiM	PSDL EN-B6 (ours)
Total	0.621	0.872	0.922	**0.943**

Table 5: Comparison with related methods over BeanTech Anomaly Detection Dataset

	AE MSE+SSIM	VT-ADL	PatchCore	PaDiM	PSDL EN-B6 (ours)
Total	0.79	0.90	0.97	**0.98**	0.97

the-art performance, even if being sub-optimal with respect to the best methods: PaDiM e PatchCore. Though, these models operate in a one-class learning scenario, differently from us.

In order to assess the effectiveness of our methodology in facing more complex situations, we tested the best model obtained from ablation studies over the modified version of Mvtec AD, Rd-MVTec AD as described in Defard et al. (2020). They applied random rotation (-10, +10 degrees) and random crop (from 256x256 to 224x224) to the train and test sets. Again, we have chosen EfficientNet-B6 architecture as the pre-trained model for the pipeline selecting layers MBConv 15, 22, 30.

In this round of experiments, seeds have been changed from run to run (for a total of 4 runs) to evaluate consistency during training time. In the same way, we changed the seeds during inference time to generate the test datasets for a total 4-vs-4-datasets scenario. Scores of Tab. 4 are aggregated by mean. Tab. 4 shows anomaly detection localization (i.e., pixel-level AUC-ROC) performance over the Rd-MVTec Dataset. Scores of other methods are taken from Defard et al. (2020). The pre-trained neural network used in PaDiM method is Wide-ResNet-50 (WR50). Same network has been adopted for SPADE. Results confirm the ability of our methodology to handle a good amount of dataset variance, reaching state-of-the-art performance. It is worth noticing that we are not using the same pre-trained network as other methods in this comparison. Nevertheless, we know from ablation studies that a Wide-ResNet-50 architecture with 1, 2, 3 layers could achieve even better scores.

3.2.2 BeanTech Anomaly Detection, Magnetic Tile Defects, and Kolektor Surface Defect Datasets

Another interesting dataset recently made available to the scientific community is the BeanTech Anomaly Detection Dataset (BTAD). It contains RGB images of three industrial products. Product 1, 2, and 3 have 400, 1000, and 399 train images, respectively. Test images are 752. A pixel-level ground truth mask is given. The experiment's settings are the same as the previous sub-section, except for pre-processing steps: images are normalized and resized but not center-cropped. Tab. 5 shows image-level detection performance over the BTAD Dataset. Scores of other methods are taken from Mishra et al. (2021), except fot PaDiM and PatchCore. For these two methods we adopted our implementation. They both used Wide-ResNet-50 as backbone network, and PatchCore subsampling level is 0.01%. We adopted such a strict level for memory reasons. The results show that our method has reached state-of-the-art methods in terms of area-under-the-precision-recall-curve (AUC-PR). Other reported methods are a classic auto-encoders trained with MSE and SSIM, and a visual-transformer network proposed in Mishra et al. (2021).

The Magnetic Tile Defects Dataset (MTD Huang et al. (2020)) is a collection of 1344 images of magnetic tiles, with five types of defects, besides normal images: Blowhole, Crack, Fray, Break, Uneven (caused by the grinding process); each image has pixel-level labels. Images are collected under multiple illumination conditions to simulate the real manufacturing process. We might expect our method to be at a disadvantage with fewer available images in this single-class scenario. Instead, Sub-section 3.1.2 has shown that with a wiser architecture choice, we could obtain state-of-the-art results. We have chosen to test our anomaly detection pipeline leveraging EfficientNet-B6 and EfficientNet-B7; we modified the pre-processing steps: images are simply normalized and resized, not center-cropped. Activation maps adopted are taken from MBConv layers 30 and 37 as done in Sub-section 3.1.2. As done in Rudolph et al. (2020), 20% of defect-free images are evaluated against at test time, with the rest used for training. Tab. 6 shows image level detection performance (AUC-ROC) over the MTD Dataset. Scores of other methods are taken from Roth et al. (2021). DifferNet method is described in Rudolph et al. (2020). PatchCore scores are the one stated by the paper, model version adopts a memory bank level subsampling of 10%. PaDiM scores come from

Table 6: Comparison with related methods over Magnetic Tile Defects Dataset

PaDiM	DifferNet	PatchCore	PSDL EN-B6 (ours)	PSDL EN-B7 (ours)
0.872	0.977	0.979	0.979	**0.988**

Table 7: Comparison with related methods over Kolektor Surface Defect Dataset

	PaDiM	PatchCore	PSDL EN-B7 (ours)
Total	0.863	0.961	**0.963**
	0.963	0.815	**0.981**

our implementation, with Wide-ResNet-50 as backbone network. The results shown confirm the strength of our approach, able to overcome the state-of-the-art performance in terms of AUC-ROC.

The Kolektor Surface Defect Dataset (KSD, Tabernik et al. (2019)) is constructed from images of defective electrical commutators surfaces with microscopic fractions or cracks. The dataset consists of 52 defective parts and 347 images with non-defective surfaces. We have chosen to test our anomaly detection pipeline leveraging EfficientNet-B7; we modified the pre-processing steps: images are first center-cropped to 1000 x 500, then resized to 500 x 2500, and then normalized. Activation maps adopted are taken from MBConv layers 37 as done in Sub-section 3.1.2. Almost 25% of defect-free images are evaluated against at test time, with the rest used for training. Tab. 7 shows image level detection AUC-ROC (first line) and pixel level (second line) over the KSD Dataset. For PaDiM and PatchCore we adopted our implementation. They both used Wide-ResNet-50 as backbone network, and PatchCore subsampling level is 0.01%. We adopted such a strict level for memory reasons. The results shown confirm the strength of our approach, able to overcome the state-of-the-art performance in terms of AUC-ROC.

3.3 Discussion

Results of experiments conducted have shown the potential of our approach in anomaly detection tasks in different contexts. The ablation studies we have conducted have given a clear insight into hyperparameter meanings and the impact of different pipeline components over the selected task. In particular, we know that the approach benefits from selecting deeper networks and activation maps with at least 200 channels without exceeding the minimum spatial resolution of 14x14.

Moreover, even if the problem has strong dimensionality constrain, i.e., the dataset should contain at least as many examples as the activation maps channels, the representation power of state-of-the-art classification networks is enough to ensure stable and optimal results. EfficientNet models, in particular, have proven to be robust and reach top performances while keeping relatively low the time needed to run the K-SVD algorithm over their activation maps. Results were in line with those presented by state-of-the-art techniques for MVTec Anomaly Detection and BTAD datasets. We also reached top performances over the Non-Aligned RD-MVTec, MTD, and KSD datasets showing robustness to increased variance and flexibility of adapting the pipeline to different contexts.

4 Conclusions

In this work, we presented a pipeline to perform anomaly detection over images. It essentially leveraged two ideas: pre-trained neural networks and a sparse learning algorithm. The methodology can thus be categorized as an unsupervised feature-based machine learning model. In comparison to similar algorithms, our is not linked to a one-class learning scenario. Moreover, the procedure has a self-learned way to select the more relevant features extracted by activation maps. The promising results we have obtained encourage us to explore further the theme of sparse learning and neural network activation maps.

The drop of a one-class-learning approach highlights the possibility of applying transfer learning procedure directly to the obtained sparse dictionaries. It might be interesting to cross-evaluate different datasets and dictionaries or combine them in a bigger pipeline. This theme is also linked to the data visualization procedure we could embed in the pipeline to better understand and visualize the learned feature. In the sparse signal process methodology, non-zero coefficients often represent key aspects of the signal that could be directly manipulated to obtain a greater contrast and highlight even better the anomalies.

REPRODUCIBILITY STATEMENT

The authors of this paper have prepared an anonymized *.zip* file containing all the scripts needed to reproduce the experiments explained in the related sections. This file has been provided as supplementary material for this submission. The authors have also planned to upload the files in a public repository once the selection process is over and the anonymity constraint is no more needed.

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11): 4311–4322, 2006. doi: 10.1109/TSP.2006.881199.

Various Authors. Pytorch, Accessed 01-August-2021. URL https://pytorch.org.

Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. *arXiv preprint arXiv:2005.02359*, 2020.

Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *CoRR*, abs/1807.02011, 2018. URL http://arxiv.org/abs/1807.02011.

Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9584–9592, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00982. URL https://ieeexplore.ieee.org/document/8954181/.

Tobias Böttger and Markus Ulrich. Real-time texture error detection on textured surfaces with compressed sensing. *Pattern Recognition and Image Analysis*, 26:88–94, 01 2016.

Tony Cai and Lie Wang. Orthogonal matching pursuit for sparse signal recovery with noise. *IEEE Trans. Inf. Theor.*, 57(7):4680–4688, July 2011. ISSN 0018-9448. doi: 10.1109/TIT.2011.2146090. URL https://doi.org/10.1109/TIT.2011.2146090.

Niv Cohen and Yedid Hoshen. Sub-Image Anomaly Detection with Deep Pyramid Correspondences. *arXiv:2005.02357 [cs]*, February 2021. URL http://arxiv.org/abs/2005.02357. arXiv: 2005.02357 version: 3.

Ingrid Daubechies, Michel Defrise, and Christine Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constrains. *Communications on Pure and Applied Mathematics*, 57, 11 2004. doi: 10.1002/cpa.20042.

Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. PaDiM: a Patch Distribution Modeling Framework for Anomaly Detection and Localization. *arXiv:2011.08785 [cs]*, November 2020. URL http://arxiv.org/abs/2011.08785. arXiv: 2011.08785.

Jia Deng, Wei Dong, Richard Socher, Lia-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Thibaud Ehret, Axel Davy, Jean-Michel Morel, and Mauricio Delbracio. Image anomalies: A review and synthesis of detection methods. *Journal of Mathematical Imaging and Vision*, 61(5):710–743, 2019.

Kjersti Engan, Sven Ole Aase, and J. Hakon Husoy. Method of optimal directions for frame design. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, volume 5, pp. 2443–2446 vol.5, 1999. doi: 10.1109/ICASSP.1999.760624.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pp. 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Yi Gu and Kang Li. A transfer model based on supervised multi-layer dictionary learning for brain tumor mri image recognition. *Frontiers in Neuroscience*, 15:550, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Fabian Herzog. SparseLandTools: A Python package for sparse representations and dictionary learning, including matching pursuit, K-SVD and applications., June 2021. URL https://doi.org/10.5281/zenodo.4916395.

Yibin Huang, Congying Qiu, and Kui Yuan. Surface defect saliency of magnetic tile. *The Visual Computer*, 36(1):85–96, 2020.

Chetak Kandaswamy, Luis Silva, Luís Alexandre, Jorge Santos, and Joaquim Sá. Improving deep neural network performance by reusing features trained with transductive transference. 09 2014. ISBN 978-3-319-11178-0. doi: 10.1007/978-3-319-11179-7_34.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. CutPaste: Self-Supervised Learning for Anomaly Detection and Localization. *arXiv:2104.04015 [cs]*, April 2021. URL http://arxiv.org/abs/2104.04015. arXiv: 2104.04015 version: 1.

Andreas Maurer, Massi Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *International conference on machine learning*, pp. 343–351. PMLR, 2013.

L. Melas-Kyriazi. Efficientnet-pytorch, Accessed 01-August-2021. URL https://github.com/lukemelas/EfficientNet-PyTorch.

Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. Vt-adl: A vision transformer network for image anomaly detection and localization. *arXiv preprint arXiv:2104.10036*, 2021.

Paolo Napoletano, Flavio Piccoli, and Raimondo Schettini. Anomaly Detection in Nanofibrous Materials by CNN-Based Self-Similarity. *Sensors (Basel, Switzerland)*, 18(1), January 2018. ISSN 1424-8220. doi: 10.3390/s18010209. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5795842/.

Marco Pimentel, David A. Clifton, Lei Clifton, and L. Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

Jonathan Pirnay and Keng Chai. Inpainting transformer for anomaly detection. 2021. URL http://arxiv.org/abs/2104.13897.

Rajat Raina, Andrew Y. Ng, and Daphne Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pp. 713–720, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143934. URL https://doi.org/10.1145/1143844.1143934.

Oliver Rippel, Patrick Mertens, and Dorit Merhof. Modeling the Distribution of Normal Data in Pre-Trained Deep Features for Anomaly Detection. *arXiv:2005.14140 [cs]*, October 2020. URL http://arxiv.org/abs/2005.14140. arXiv: 2005.14140.

Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. *arXiv preprint arXiv:2106.08265*, 2021.

Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same Same But DifferNet: Semi-Supervised Defect Detection with Normalizing Flows. *arXiv:2008.12577 [cs, eess]*, August 2020. URL http://arxiv.org/abs/2008.12577. arXiv: 2008.12577 version: 1.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In Marc Niethammer, Martin Styner, Stephen Aylward, Hongtu Zhu, Ipek Oguz, Pew-Thian Yap, and Dinggang Shen (eds.), *Information Processing in Medical Imaging*, pp. 146–157, Cham, 2017. Springer International Publishing. ISBN 978-3-319-59050-9.

Hoo-Chang Shin, H. Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, D. Mollura, and R. Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, 35:1285–1298, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Domen Tabernik, Samo Šela, Jure Skvarč, and Danijel Skočaj. Segmentation-Based Deep-Learning Approach for Surface-Defect Detection. *Journal of Intelligent Manufacturing*, May 2019. ISSN 1572-8145. doi: 10.1007/s10845-019-01476-x.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR, 2019.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2820–2828, 2019.

Hao Tang, Hong Liu, Wei Xiao, and Nicu Sebe. When dictionary learning meets deep learning: Deep dictionary learning and coding network for image recognition with limited data. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):2129–2141, 2020.

Guodong Wang, Shumin Han, Errui Ding, and Di Huang. Student-teacher feature pyramid matching for unsupervised anomaly detection. *CoRR*, abs/2103.04257, 2021. URL https://arxiv.org/abs/2103.04257.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

Jie Yang, Yong Shi, and Zhiquan Qi. DFR: Deep Feature Reconstruction for Unsupervised Anomaly Segmentation. *arXiv:2012.07122 [cs]*, December 2020. URL http://arxiv.org/abs/2012.07122. arXiv: 2012.07122 version: 1.

Jihun Yi and Sungroh Yoon. Patch SVDD: Patch-level SVDD for Anomaly Detection and Segmentation. *arXiv:2006.16067 [cs]*, July 2020. URL http://arxiv.org/abs/2006.16067. arXiv: 2006.16067.

John Zarka, Louis Thiry, Tomás Angles, and Stéphane Mallat. Deep network classification by scattering and homotopy dictionary learning. *arXiv preprint arXiv:1910.03561*, 2019.

Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, 112:107706, April 2021. ISSN 00313203. doi: 10.1016/j.patcog.2020.107706. URL https://linkinghub.elsevier.com/retrieve/pii/S0031320320305094.

A Appendix

We include in this appendix a detailed description of the ablation studies conducted to understand the impact of the dictionary initialization, the number of iterations, and the sparsity level of the K-SVD algorithm. These studies have given us a clear understanding of how to execute the sparse dictionary learning phase, and the best hyperparameters have been adopted in all the other scenarios. After each experiment, the best settings are carried on during subsequent explorations in a cascade approach.

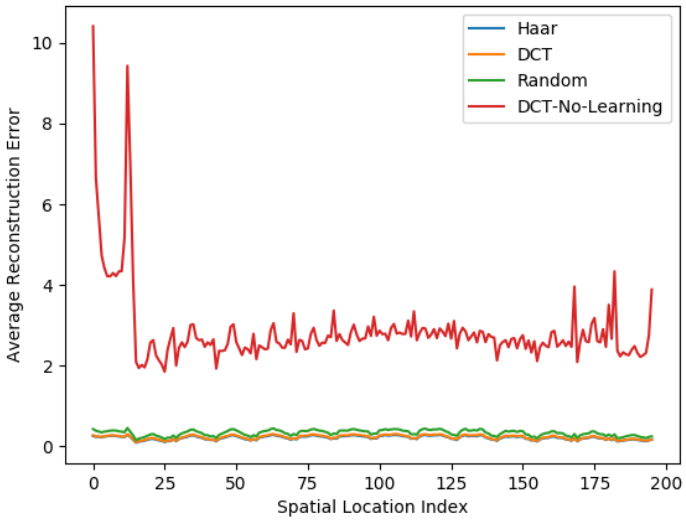


Figure 2: The graph shows the average reconstruction error for each spatial location of the activation map, over the training dataset. With the settings of experiments of subsection A.1, the learned dictionaries have a lower reconstruction error than the fixed DCT dictionary.

A.1 Dictionary Selection

The first aspect we have investigated is whether the dictionary learning procedure is relevant in the pipeline. Activation maps considered are the output of the third ResNet-18 block (layer 3). We have selected this layer because it achieves a good trade-off between depth and spatial resolution: it is well proven that layers close to the input generate activation maps with small receptive fields that can spot coarse features, while layers close to the output have bigger receptive fields and spot detailed features.

To show the importance of the learning procedure, we have run the pipeline with the dictionaries initialized in different ways (but always consistently inside the same run) and with a fixed dictionary based on Discrete Cosine Transform basis. In the latter case, the dictionary is never updated; we run the OMP algorithm to obtain the sparse representation of the selected activation maps. In the remaining cases, we have again chosen the DCT, Haar, and random initialization for the K-SVD algorithm.

Dictionaries are kept squared (256x256), and they are 196 (one for each spatial location). Orthogonal Matching Pursuit is run until we reach a 0-sparsity level of 64 (which is a quarter of the dictionary dimension). K-SVD algorithm is run for 20 iterations. If not stated otherwise, this is the standard configuration of every experiment.

Tab. 8 shows the results for each class of MVTec test dataset. The first number in each row is the per-image AUC-ROC, while the second is the per-pixel AUC-ROC. We can see that the learning procedure is fundamental to obtain relevant scores. Without the learning step, the process of identifying anomalies is barely random. It is also interesting to note that different initializations do not have a relevant impact on pipeline performance.

We have also investigated the mean reconstruction error for a specific patch of the training dataset. Fig. 2 shows a graph with the average mean squared error per each spatial location. The non-learning procedure has a higher reconstruction error (one order magnitude) over each spatial location, while the learned dictionaries reach a significantly lower error.

We have decided to keep the DCT basis as dictionary initialization for the slightly better results in terms of the total per-pixel score for successive experiments.

A.2 K-SVD Iterations

Previous experiments have shown a correlation between the lower reconstruction error and better performances over the test set. This has led us to explore the number of K-SVD iterations we can use.

Table 8: Dictionary selection results

	DCT No-Learning	DCT	Haar	Random
Textures classes	0.6742	**0.9486**	0.9568	0.947
	0.527	0.9304	**0.9326**	0.9278
Objects classes	0.4978	0.9134	0.9092	**0.921**
	0.551	**0.9716**	0.9674	0.9642
Total	0.586	0.931	0.933	**0.934**
	0.539	**0.951**	0.950	0.946

Table 9: Iterations selection results

	10 iterations	20 iterations	50 iterations
Textures classes	**0.9512**	0.9486	0.9472
	0.929	0.9304	**0.9342**
Objects classes	0.9128	**0.9134**	0.9088
	0.965	**0.9716**	0.9678
Total	**0.932**	0.931	0.928
	0.947	**0.951**	**0.951**

We have tested the pipeline with 10, 20, and 50 as iteration numbers and reported results in Tab. 9. Again, the first number in each row is the per-image AUC-ROC, while the second is the per-pixel AUC-ROC. Average reconstruction error per patch is shown in Fig. 3(a). In this case, the dynamic over each patch is better shown thanks to the homogeneous scale of the plot. This particular outline is given because the spatial error map (size 14x14) is unrolled. Fig. 3(b) shows the surface plot for 50 iterations run. As expected, the reconstruction error is higher in the central area of the image, where the information is more complex, while on the borders is relatively smaller.

We can see that the reconstruction error decreases if the iterations number is higher, though this is not directly reflected in a better AUC-ROC score (both per-images and per-pixels). This probably reflects the well-known behavior of classic anomaly detection reconstruction methods: models able to reconstruct signals with higher accuracy are also inclined to reconstruct anomalies. Since our target is mainly anomaly segmentation (i.e., per-pixels score), 20 iterations are the best fit for our experiments, speeding up the process with respect to using more iterations.

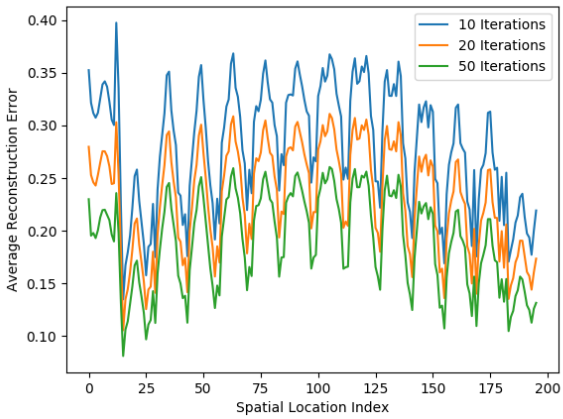
A.3 Sparsity Level

This last section investigates the level of sparsity to be adopted in the OMP algorithm. The pipeline configuration is mutated from previous experiments (A.1 and A.2). We only let the sparsity level vary: 64, 128, 192. These values correspond to a quarter, a half, and a three-quarter of the total number of dictionary elements.

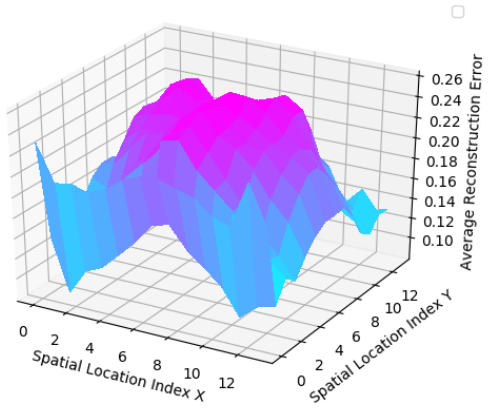
Tab. 10 presents the results. Interestingly, the best scores in terms of AUC-ROC are achieved without doubt, with 64 as sparsity level. Performance significantly degrades as the sparsity level increases.

To better understand the motivation of these decreasing scores, we plot the average reconstruction error at each spatial location for the algorithm run with 64 and 192 as sparsity levels. As we can see in Fig. 4, the dictionary learned through a less strict procedure generates again a much lower reconstruction error, though this is not reflected in a better score. We claim again that too powerful reconstruction models suffer from also reconstructing anomalies.

This intuition is confirmed by the analysis we have carried out over the test set, plotting the average reconstruction error for both experiments: the one with 64 and the one with 192. In Fig. 5, we can see that the reconstruction error, in the second case, pushes the two distributions (the one over normal examples and the one over anomalous example) to be closer. This is a signal of training set reconstruction overfitting. The result is a threshold value not able to split defective patches from non-defective ones, downgrading AUC-ROC performance.



(a)



(b)

Figure 3: The graph on the top shows the average reconstruction error for each spatial location of the activation map, over the training dataset. With the settings of experiments of subsection A.2, the higher the number of iterations the lower the reconstruction error. The surface plot on the bottom shows the same average reconstruction error for 50 iterations run only. Spatial dimension are here preserved.

Table 10: Sparsity level selection results

	64	**128**	**192**
Textures classes	**0.9486**	0.892	0.6846
	0.9304	0.8662	0.4952
Objects classes	**0.9134**	0.836	0.6094
	0.9716	0.9258	0.7168
Total	**0.931**	0.864	0.647
	0.951	0.896	0.606

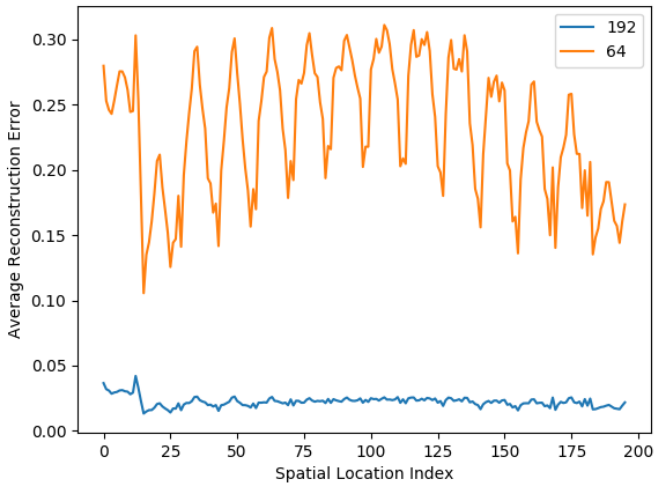
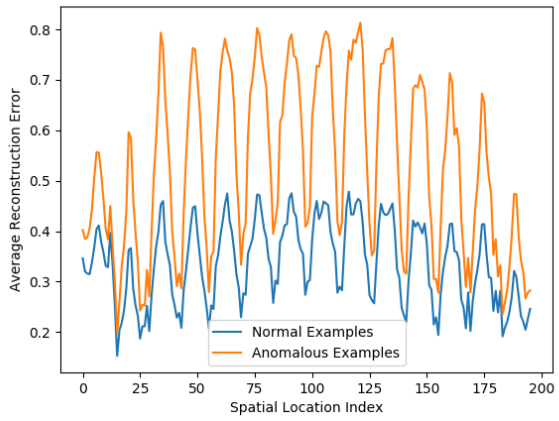
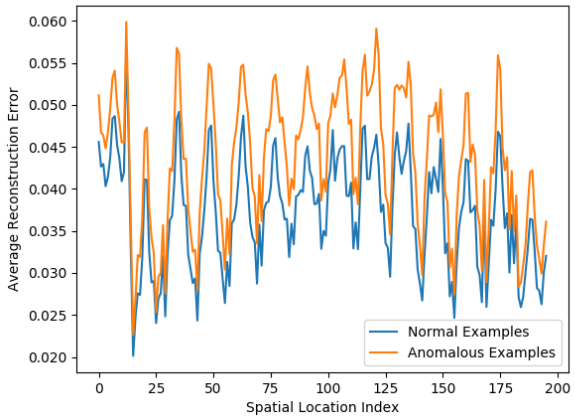


Figure 4: The graph shows the average reconstruction error for each spatial location of the activation map, over the training dataset. The increased sparsity level is causing a higher reconstruction error.



(a)



(b)

Figure 5: The graph on the top shows the average reconstruction error for each spatial location of the activation map over the test set, for the experiment run with 64 as sparsity level. The graph on the bottom shows the same error for the experiment run with 192 as sparsity level.